PyPalEx

1.3.1

Generated by Doxygen 1.9.5

1 PyPalEx: The Python Palette Extractor

1.1 Description

PyPalEx is a tool for extracting color palettes from images and generating a JSON format file with light and dark color themes. This tool is intended to be OS independent, for use by the tech community for developing their own custom theme managers or by artists who want to extract color palettes for their art from images, pictures or wallpapers they adore.

2 Namespace Index

2.1 Package List

Here are the packages with brief descriptions (if available):

pypalex **Python Palette Extractor: extracts color palettes from images**	**2**
pypalex.__main__	**3**
pypalex.arg_messages	**7**
pypalex.constants	**8**
pypalex.conversion_utils	**12**
pypalex.extraction_utils	**15**
pypalex.Extractor	**22**

3 Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

4 File Index

4.1 File List

Here is a list of all files with brief descriptions:

5 Namespace Documentation

5.1 pypalex Namespace Reference

Python Palette Extractor: extracts color palettes from images.

Namespaces

- namespace __main__
- namespace arg_messages
- namespace constants
- namespace conversion_utils
- namespace extraction_utils
- namespace Extractor
- namespace image_utils

5.1.1 Detailed Description

Python Palette Extractor: extracts color palettes from images.

PyPalEx is a tool for extracting color palettes from images and generating a JSON format file with light and dark color themes. This tool is intended to be OS independent, for use by the tech community for developing their own custom theme managers or by artists who want to extract color palettes for their art from images, pictures or wallpapers they adore.

5.2 pypalex.__main__ Namespace Reference

Functions

- def main ()

  Main script function.
- def handle_args ()

  Handles the arguments passed to PyPalEx.
- def extract_color_palettes ()

  Handles color extraction from image(s).
- def setup_argument_parser ()

  Sets up the argument parser for command line arguments.
- def check_sources (filepaths, path=None)

  Checks each of the sources provided and removes any bad sources.
- def check_path (path)

  Check the path to make sure it exists.
- def set_global_args (args)

  Sets the global variables using the arguments.
- def check_source (filepath)

  Checks to make sure the path leads to a file.

Variables

- list EXTRACTORS = [ ]

  List of Extractor class objects for each individual image.
- list PROPER_IMAGES = [ ]

  List of real/existing image file path(s).
- list FILENAMES = [ ]

  List of image filenames.
- list OUTPUT_FILEPATHS = [ ]

  List of output file path(s) for each image.
- string OUTPUT_PATH = ''

  The path to the output directory where all JSON files will be saved.
- string OUTPUT_TAIL = "-color_palette.json"

  The tail to append to each output filepath.
- bool PASTEL_L = False

  Flag to convert light color palette to pastel.
- bool PASTEL_N = False

  Flag to convert normal color palette to pastel.
- bool PASTEL_D = False

  Flag to convert dark color palette to pastel.
- bool SAT_PREF_L = False

  Flag that gives preference to more saturated colors of the light color palette.
- bool SAT_PREF_N = False

  Flag that gives preference to more saturated colors of the normal color palette.
- bool SAT_PREF_D = False

  Flag that gives preference to more saturated colors of the dark color palette.

5.2.1 Function Documentation

5.2.1.1 check_path()

```
def check_path (
        path )
```

Check the path to make sure it exists.

Parameters

path	The path to a directory.

Returns

True if the path exists and is not a file, False otherwise.

5.2.1.2 check_source()

```
def check_source (
        filepath )
```

Checks to make sure the path leads to a file.

Parameters

filepath	Path to file with filename and file extension.

Returns

True if file exists, False otherwise.

5.2.1.3 check_sources()

```
def check_sources (
        filepaths,
        path = None )
```

Checks each of the sources provided and removes any bad sources.

Any filepaths or source files that are not images or do not exist get removed.

Parameters

filepaths	List of file paths.
path	A path to the images, if it is provided.

Returns

True if all/some sources are good, False if all sources are bad.

5.2.1.4 extract_color_palettes()

```
def extract_color_palettes ( )
```

Handles color extraction from image(s).

5.2.1.5 handle_args()

```
def handle_args ( )
```

Handles the arguments passed to PyPalEx.

5.2.1.6 main()

```
def main ( )
```

Main script function.

5.2.1.7 set_global_args()

```
def set_global_args (
        args )
```

Sets the global variables using the arguments.

Parameters

args	User-supplied arguments.

5.2.1.8 setup_argument_parser() `def setup_argument_parser ( )`

Sets up the argument parser for command line arguments.

Returns

A command line argument parsing object.

5.2.2 Variable Documentation

5.2.2.1 EXTRACTORS `list EXTRACTORS = [ ]`

List of Extractor class objects for each individual image.

5.2.2.2 FILENAMES `list FILENAMES = [ ]`

List of image filenames.

5.2.2.3 OUTPUT_FILEPATHS `list OUTPUT_FILEPATHS = [ ]`

List of output file path(s) for each image.

5.2.2.4 OUTPUT_PATH `string OUTPUT_PATH = ''`

The path to the output directory where all JSON files will be saved.

5.2.2.5 OUTPUT_TAIL `string OUTPUT_TAIL = "-color_palette.json"`

The tail to append to each output filepath.

5.2.2.6 PASTEL_D `bool PASTEL_D = False`

Flag to convert dark color palette to pastel.

5.2.2.7 PASTEL_L `bool PASTEL_L = False`

Flag to convert light color palette to pastel.

5.2.2.8 PASTEL_N `bool PASTEL_N = False`

Flag to convert normal color palette to pastel.

5.2.2.9 PROPER_IMAGES `list PROPER_IMAGES = [ ]`

List of real/existing image file path(s).

5.2.2.10 SAT_PREF_D `bool SAT_PREF_D = False`

Flag that gives preference to more saturated colors of the dark color palette.

5.2.2.11 SAT_PREF_L `bool SAT_PREF_L = False`

Flag that gives preference to more saturated colors of the light color palette.

5.2.2.12 SAT_PREF_N `bool SAT_PREF_N = False`

Flag that gives preference to more saturated colors of the normal color palette.

5.3 pypalex.arg_messages Namespace Reference

Functions

- def bad_source_message ()

  Generates an error message if the sources provided were not images.
- def bad_path_message ()

  Generates an error message if the directory provided is not a valid directory.
- def no_args_help_message ()

  Generates a help message if no arguments were presented.

5.3.1 Function Documentation

5.3.1.1 bad_path_message() `def bad_path_message ( )`

Generates an error message if the directory provided is not a valid directory.

Returns

The "bad directory" message.

5.3.1.2 bad_source_message() `def bad_source_message ( )`

Generates an error message if the sources provided were not images.

Returns

The "bad sources" message.

5.3.1.3 no_args_help_message() `def no_args_help_message ( )`

Generates a help message if no arguments were presented.

Returns

The "no arguments" help message.

5.4 pypalex.constants Namespace Reference

Variables

- list BLACK_RGB = [0, 0, 0]
- list WHITE_RGB = [255, 255, 255]
- list RED_RGB = [255, 0, 0]
- list YELLOW_RGB = [255, 234, 0]
- list GREEN_RGB = [0, 255, 0]
- list CYAN_RGB = [0, 255, 255]
- list BLUE_RGB = [0, 0, 255]
- list MAGENTA_RGB = [255, 0, 255]
- int BLACK_HEX = 0x000000
- int WHITE_HEX = 0xFFFFFF
- int RED_HEX = 0xFF0000
- int YELLOW_HEX = 0xFFEA00
- int GREEN_HEX = 0x00FF00
- int CYAN_HEX = 0x00FFFF

- int BLUE_HEX = 0x0000FF
- int MAGENTA_HEX = 0xFF00FF
- int RED_HUE = 0
- int YELLOW_HUE = 55
- int GREEN_HUE = 120
- int CYAN_HUE = 180
- int BLUE_HUE = 240
- int MAGENTA_HUE = 300
- list RED_HUE_RANGE_MAX = [330, 360]
- list RED_HUE_RANGE_MIN = [0, 25]
- list YELLOW_HUE_RANGE = [25, 64]
- list GREEN_HUE_RANGE = [64, 170]
- list CYAN_HUE_RANGE = [170, 210]
- list BLUE_HUE_RANGE = [210, 260]
- list MAGENTA_HUE_RANGE = [260, 330]
- list BLACK_BRIGHTNESS_RANGE = [0.0, 50.0]
- list GRAY_BRIGHTNESS_RANGE = [50.0, 75.0]
- list WHITE_BRIGHTNESS_RANGE = [75.0, 100.0]
- list SATURATION_RANGE = [5.0, 100.0]
- list BRIGHTNESS_RANGE = [25.0, 100.0]
- list PASTEL_SATURATION_RANGE = [15.0, 75.0]
- list PASTEL_BRIGHTNESS_RANGE = [50.0, 100.0]

5.4.1 Variable Documentation

5.4.1.1 BLACK_BRIGHTNESS_RANGE `list BLACK_BRIGHTNESS_RANGE = [0.0, 50.0]`

5.4.1.2 BLACK_HEX `int BLACK_HEX = 0x000000`

5.4.1.3 BLACK_RGB `list BLACK_RGB = [0, 0, 0]`

5.4.1.4 BLUE_HEX `int BLUE_HEX = 0x0000FF`

5.4.1.5 BLUE_HUE `int BLUE_HUE = 240`

5.4.1.6 BLUE_HUE_RANGE `list BLUE_HUE_RANGE = [210, 260]`

5.4.1.7 BLUE_RGB `list BLUE_RGB = [0, 0, 255]`

5.4.1.8 BRIGHTNESS_RANGE `list BRIGHTNESS_RANGE = [25.0, 100.0]`

5.4.1.9 CYAN_HEX `int CYAN_HEX = 0x00FFFF`

5.4.1.10 CYAN_HUE `int CYAN_HUE = 180`

5.4.1.11 CYAN_HUE_RANGE `list CYAN_HUE_RANGE = [170, 210]`

5.4.1.12 CYAN_RGB `list CYAN_RGB = [0, 255, 255]`

5.4.1.13 GRAY_BRIGHTNESS_RANGE `list GRAY_BRIGHTNESS_RANGE = [50.0, 75.0]`

5.4.1.14 GREEN_HEX `int GREEN_HEX = 0x00FF00`

5.4.1.15 GREEN_HUE `int GREEN_HUE = 120`

5.4.1.16 GREEN_HUE_RANGE `list GREEN_HUE_RANGE = [64, 170]`

5.4.1.17 GREEN_RGB `list GREEN_RGB = [0, 255, 0]`

5.4.1.18 MAGENTA_HEX `int MAGENTA_HEX = 0xFF00FF`

5.4.1.19 MAGENTA_HUE `int MAGENTA_HUE = 300`

5.4.1.20 MAGENTA_HUE_RANGE `list MAGENTA_HUE_RANGE = [260, 330]`

5.4.1.21 MAGENTA_RGB `list MAGENTA_RGB = [255, 0, 255]`

5.4.1.22 PASTEL_BRIGHTNESS_RANGE `list PASTEL_BRIGHTNESS_RANGE = [50.0, 100.0]`

5.4.1.23 PASTEL_SATURATION_RANGE `list PASTEL_SATURATION_RANGE = [15.0, 75.0]`

5.4.1.24 RED_HEX `int RED_HEX = 0xFF0000`

5.4.1.25 RED_HUE `int RED_HUE = 0`

5.4.1.26 RED_HUE_RANGE_MAX `list RED_HUE_RANGE_MAX = [330, 360]`

5.4.1.27 RED_HUE_RANGE_MIN `list RED_HUE_RANGE_MIN = [0, 25]`

5.4.1.28 RED_RGB `list RED_RGB = [255, 0, 0]`

5.4.1.29 SATURATION_RANGE `list SATURATION_RANGE = [5.0, 100.0]`

5.4.1.30 WHITE_BRIGHTNESS_RANGE `list WHITE_BRIGHTNESS_RANGE = [75.0, 100.0]`

5.4.1.31 WHITE_HEX `int WHITE_HEX = 0xFFFFFF`

5.4.1.32 WHITE_RGB `list WHITE_RGB = [255, 255, 255]`

5.4.1.33 YELLOW_HEX `int YELLOW_HEX = 0xFFEA00`

5.4.1.34 YELLOW_HUE `int YELLOW_HUE = 55`

5.4.1.35 YELLOW_HUE_RANGE `list YELLOW_HUE_RANGE = [25, 64]`

5.4.1.36 YELLOW_RGB `list YELLOW_RGB = [255, 234, 0]`

5.5 pypalex.conversion_utils Namespace Reference

Functions

- def rgb_to_hsv (rgb_array)
  Converts RGB array [r,g,b] to HSV array [h,s,v].
- def hsv_to_hex (hsv_array)
  Convert HSV array [h,s,v] to HEX string 'ffffff'.
- def hsv_to_rgb (hsv_array)
  Convert HSV array [h,s,v] to RGB array [r,g,b].
- def rgb_to_hex (rgb_array)
  Convert RGB array [r,g,b] to HEX string 'ffffff'.

5.5.1 Function Documentation

5.5.1.1 hsv_to_hex()

```
def hsv_to_hex (
        hsv_array )
```

Convert HSV array [h,s,v] to HEX string 'ffffff'.

HSV where h is in the set [0, 359] and s, v are in the set [0.0, 100.0]. HEX string is in the set ["000000", "ffffff"].

Parameters

hsv_array	HSV array [h,s,v].

Returns

A HEX string.

5.5.1.2 hsv_to_rgb()

```
def hsv_to_rgb (
    hsv_array )
```

Convert HSV array [h,s,v] to RGB array [r,g,b].

HSV where h is in the set [0, 359] and s, v are in the set [0.0, 100.0]. RGB where [r,g,b] are in the set [0, 255]. Formula adapted from `https://www.rapidtables.com/convert/color/hsv-to-rgb.html`

Parameters

hsv_array	HSV array [h,s,v].

Returns

RGB array [r,g,b].

5.5.1.3 rgb_to_hex()

```
def rgb_to_hex (
    rgb_array )
```

Convert RGB array [r,g,b] to HEX string 'ffffff'.

RGB where [r,g,b] are in the set [0, 255]. HEX string is in the set ["000000", "ffffff"].

Parameters

rgb_array	RGB array [r,g,b].

Returns

A HEX string.

5.5.1.4 rgb_to_hsv()

```
def rgb_to_hsv (
    rgb_array )
```

Converts RGB array [r,g,b] to HSV array [h,s,v].

RGB where [r,g,b] are in the set [0, 255]. HSV where h is in the set [0, 359] and s, v are in the set [0.0, 100.0]. Formula adapted from `https://www.rapidtables.com/convert/color/rgb-to-hsv.html`

Parameters

rgb_array	RGB array [r,g,b].

Returns

HSV array [h,s,v].

5.6 pypalex.extraction_utils Namespace Reference

Functions

- def extract_ratios (hsv_img_matrix_2d)

  Extracts the ratios of hues per pixel.
- def construct_base_color_dictionary (hsv_img_matrix_2d)

  Constructs dictionary of base colors from an array of HSV pixel values.
- def extract_color_palettes (base_color_dict, sat_pref_list)

  Extracts dominant light, normal, dark color palettes from each of the base colors.
- def check_missing_colors (base_color_dict, extracted_colors_dict)

  Checks for any missing colors in the base color dictionary and borrows them from the surrounding colors.
- def generate_remaining_colors (extracted_colors_dict, ratios)

  Generate the remaining black and white, and background and foreground colors.
- def extract_color_types (hsv_base_color_matrix_and_sat_prefs)

  Extracts the dominant color types from a base color.
- def get_left_and_right_colors (origin_color_name)

  Gets the color names of the colors that are to the left and right of the originating color.
- def borrow_color (extracted_colors_dict, origin, borrow_left, borrow_right)

  Borrows a color from one of the extracted color types of the base colors.
- def get_dominant_hue (extracted_colors_dict, ratios)

  Calculates the dominant hue.
- def generate_black_and_white (dominant_hue)

  Generates black and white color types using the dominant hue.
- def generate_background_and_foreground (dominant_hue, complementary_hue)

  Generates the background and foreground colors.
- def sort_by_bright_value (hsv_base_color_matrix)

  Sorts the colors by the brightness value.
- def extract_dominant_color (hsv_color_type_matrix, sat_pref)

  Extracts the dominant color from a color type.
- def check_missing_color_types (light_color, norm_color, dark_color)

  Checks to make sure all the color types have been properly set.
- def check_sat_and_bright (hsv_color)

  Normalize saturation and brightness value.
- def calculate_centroid (hsv_color_type_matrix)

  Calculates the centroid for a color type.
- def find_closest_to_centroid (hsv_color_type_matrix, centroid, sat_pref)

  Finds a color from a color type that is closest to the centroid.

5.6.1 Function Documentation

5.6.1.1 borrow_color()

```
def borrow_color (
        extracted_colors_dict,
        origin,
        borrow_left,
        borrow_right )
```

Borrows a color from one of the extracted color types of the base colors.

Parameters

extracted_colors_dict	A Dictionary of extracted colors.
origin	The name of the originating color.
borrow_left	The name of the color to borrow from, to the left of origin.
borrow_right	The name of the color to borrow from, to the right of origin.

Returns

A numpy array of a borrowed color.

5.6.1.2 calculate_centroid()

```
def calculate_centroid (
        hsv_color_type_matrix )
```

Calculates the centroid for a color type.

The centroid is basically the average color of a set of colors in [h,s,v] format. The centroid is a point in 3-dimensional space. The following sources were used to make this algorithm: `http://mkweb.bcgsc.↵ca/color-summarizer/?faq#averagehue` and `https://stackoverflow.com/a/8170595/17047816`

Parameters

hsv_color_type_matrix	A 2D numpy array of a color type in [h,s,v] format.

Returns

List of centroid color values in [h,s,l] format.

5.6.1.3 check_missing_color_types()

```
def check_missing_color_types (
        light_color,
        norm_color,
        dark_color )
```

Checks to make sure all the color types have been properly set.

If a color type is missing, then it will be derived from the existing color types.

Parameters

light_color	A numpy array of a light color type in [h,s,v] format.
norm_color	A numpy array of a normal color type in [h,s,v] format.
dark_color	A numpy array of a dark color type in [h,s,v] format.

5.6.1.4 check_missing_colors()

```
def check_missing_colors (
        base_color_dict,
        extracted_colors_dict )
```

Checks for any missing colors in the base color dictionary and borrows them from the surrounding colors.

Parameters

base_color_dict	A dictionary of 2D numpy arrays for each of the base colors.
extracted_colors_dict	A Dictionary of extracted colors.

5.6.1.5 check_sat_and_bright()

```
def check_sat_and_bright (
        hsv_color )
```

Normalize saturation and brightness value.

The normalization process is to make sure that colors are visible, distinguishable and tolerable to look at. These ranges for saturation and brightness values are defined in constants.py. This step can be removed if it is not needed as it does not impact the extraction process.

Parameters

hsv_color	A numpy array of a color type in [h,s,v] format.

5.6.1.6 construct_base_color_dictionary()

```
def construct_base_color_dictionary (
        hsv_img_matrix_2d )
```

Constructs dictionary of base colors from an array of HSV pixel values.

Base colors are classified as [red, yellow, green, cyan, blue, magenta].

Parameters

hsv_img_matrix_2d	A 2D numpy array of pixels from an image, in [h,s,v] format.

Returns

Dictionary of base colors.

5.6.1.7 extract_color_palettes()

```
def extract_color_palettes (
        base_color_dict,
        sat_pref_list )
```

Extracts dominant light, normal, dark color palettes from each of the base colors.

Parameters

base_color_dict	A dictionary of 2D numpy arrays for each of the base colors.
sat_pref_list	List of saturation preference flags for light, normal, dark color palettes.

Returns

Dictionary of light, normal, dark color palettes for each of the base colors.

5.6.1.8 extract_color_types()

```
def extract_color_types (
        hsv_base_color_matrix_and_sat_prefs )
```

Extracts the dominant color types from a base color.

A color type is either a light, normal, or dark version of a base color.

Parameters

hsv_base_color_matrix_and_sat_prefs	A tuple of a 2D numpy array of a base color in [h,s,v] format and a list of saturation preference flags for light, normal, dark color palettes.

Returns

List of dominant numpy array color types in [h,s,v] format.

5.6.1.9 extract_dominant_color()

```
def extract_dominant_color (
        hsv_color_type_matrix,
        sat_pref )
```

Extracts the dominant color from a color type.

A color type is either a light, normal, or dark version of a base color.

Parameters

hsv_color_type_matrix	A 2D numpy array of a color type in [h,s,v] format. @para sat_pref A saturation preference flag for one of the light, normal, dark color palettes.

Returns

A numpy array of a dominant color from a color type in [h,s,v] format.

5.6.1.10 extract_ratios()

```
def extract_ratios (
            hsv_img_matrix_2d )
```

Extracts the ratios of hues per pixel.

Parameters

hsv_img_matrix_2d	A 2D numpy array of pixels from an image in [h,s,v] format.

Returns

Dictionary of hue ratios (percentage) in set [0.0, 100.0]

5.6.1.11 find_closest_to_centroid()

```
def find_closest_to_centroid (
            hsv_color_type_matrix,
            centroid,
            sat_pref )
```

Finds a color from a color type that is closest to the centroid.

The distance between the centroid color and each of the other individual colors is calculated in 3-dimensional space using the Euclidean Distance formula from the following sources: https://stackoverflow.↵
com/a/35114586/17047816 and https://byjus.com/maths/distance-between-two-points-3d/. Preference is given to saturated color when the sat_pref parameter is set by using a parabola formula: f(x) = ((x - point_of_symmetry_of_parabola) / 6) ^ 2 where 0 <= x <= 100. In this formula, x is our current saturation and point_of_symmetry_of_parabola is our preferred saturation. The source used to find and graph this formula: https://www.graphfree.com/grapher.html

Note

Possible feature addition in the future, where the user can have the option to input their preferred saturation (e.g. pref_sat) and it can be used to replace the point_of_symmetry_of_parabola in the parabola formula. If saturation is preferred (e.g. sat_pref) but no preferred saturation is set by the user, then the default should be 60. And if saturation is not preferred, then that value should be set to None.

Parameters

hsv_color_type_matrix	A 2D numpy array of a color type in [h,s,v] format.
centroid	List of centroid color values in [h,s,l] format. @para sat_pref A saturation preference flag for one of the light, normal, dark color palettes.

Returns

List of all the colors in [h,s,v] format that are the shortest distance away from the centroid.

5.6.1.12 generate_background_and_foreground()

```
def generate_background_and_foreground (
        dominant_hue,
        complementary_hue )
```

Generates the background and foreground colors.

The background and foreground colors are based on the dominant hue in an image and it's complimentary hue. The saturation and brightness values for the background and foreground colors need to be hardcoded to be easier to look at.

Parameters

dominant_hue	The dominant hue of an image.
complementary_hue	The complimentary hue to the dominant hue.

Returns

Numpy array of light and dark background and foreground colors in [h,s,v] format.

5.6.1.13 generate_black_and_white()

```
def generate_black_and_white (
        dominant_hue )
```

Generates black and white color types using the dominant hue.

The saturation and brightness values, for the black and white color types, needs to be hardcoded in order to not interfere with the background and foreground colors.

Parameters

dominant_hue	The dominant hue of an image.

Returns

List of black and white color types in [h,s,v] format.

5.6.1.14 generate_remaining_colors() `def generate_remaining_colors ( extracted_colors_dict, ratios )`

Generate the remaining black and white, and background and foreground colors.

Parameters

extracted_colors_dict	A Dictionary of extracted colors.
ratios	A Dictionary of ratios of the base colors in the image.

5.6.1.15 get_dominant_hue() `def get_dominant_hue ( extracted_colors_dict, ratios )`

Calculates the dominant hue.

The dominant hue, also referred to as the average hue, is based on the color ratios and the colors extracted from an image.

Parameters

extracted_colors_dict	A Dictionary of extracted colors.
ratios	A Dictionary of ratios of the base colors in the image.

Returns

The dominant hue in an image.

5.6.1.16 get_left_and_right_colors() `def get_left_and_right_colors ( origin_color_name )`

Gets the color names of the colors that are to the left and right of the originating color.

There are two ways to think about left and right on a color wheel: from the inside looking outward and from the outside looking inward. This has an effect on how we think of the linear format of the color wheel. For this package we will think about left and right colors using the latter option.

Parameters

origin_color_name	The name of the originating color.

Returns

List of color names that are to the left and right of the originating color.

5.6.1.17 sort_by_bright_value() `def sort_by_bright_value ( hsv_base_color_matrix )`

Sorts the colors by the brightness value.

A color type is either a light, normal, or dark version of a base color.

Parameters

hsv_base_color_matrix	A 2D numpy array of a base color in [h,s,v] format.

Returns

List of numpy array color types in [h,s,v] format.

5.7 pypalex.Extractor Namespace Reference

Classes

- class Extractor

  Extracts colors given a matrix of HSV values extracted from an image.

5.8 pypalex.image_utils Namespace Reference

Functions

- def process_image (image)

  Processes PIL Image object.
- def save_palette_to_file (color_palette, output_filepath)

  Saves color palette to json file.
- def rescale_image (image)

  Rescales image to a smaller sampling size.
- def process_helper (rgb_matrix_2d)

  Helper function for multiprocessing conversion operations.

5.8.1 Function Documentation

5.8.1.1 process_helper() `def process_helper ( rgb_matrix_2d )`

Helper function for multiprocessing conversion operations.

Helps convert from [r,g,b] to [h,s,v].

Parameters

rgb_matrix_2d	A 2D matrix of rgb values.

Returns

A numpy array/2D matrix of converted [h,s,v] values.

5.8.1.2 process_image()

```
def process_image (
        image )
```

Processes PIL Image object.

Multiprocessing example from: https://stackoverflow.com/a/45555516

Parameters

image	PIL Image object.

Returns

2D numpy array of [h,s,v] arrays (pixels) from image.

5.8.1.3 rescale_image()

```
def rescale_image (
        image )
```

Rescales image to a smaller sampling size.

Parameters

image	PIL Image object.

Returns

Tuple of the new width and height of image.

5.8.1.4 save_palette_to_file()

```
def save_palette_to_file (
        color_palette,
        output_filepath )
```

Saves color palette to json file.

If a file with the same name already exists, it is overwritten.

Parameters

color_palette	Dictionary of light, normal, and dark color palettes.
output_filepath	Output file path with filename of where to store color palette.

6 Class Documentation

6.1 Extractor Class Reference

Extracts colors given a matrix of HSV values extracted from an image.

Public Member Functions

- def __init__ (self, hsv_img_matrix_2d, output_filepath, pastel_light=False, pastel_normal=False, pastel_dark=False, sat_pref_light=False, sat_pref_normal=False, sat_pref_dark=False)

  Extractor Constructor.
- def run (self)

  Main method for Extractor class.
- def check_pastel_conversion (self)

  Checks to see if any of the palettes should be converted to pastel.
- def construct_scheme_dictionary (self)

  Constructs a dictionary of color schemes by combining color palettes.
- def convert_pastel_light (self)

  Converts light palette to pastel.
- def convert_pastel_normal (self)

  Converts normal palette to pastel.
- def convert_pastel_dark (self)

  Converts dark palette to pastel.
- def convert_pastel (self, hsv_color)

  Converts/normalizes HSV color to pastel.

Public Attributes

- hsv_img_matrix_2d

  A 2D numpy array of pixels from an image in [h,s,v] format.
- output_filepath

  Output file path with filename of where to store color palette.
- pastel_light

  Flag to convert light color palette to pastel.
- pastel_normal

  Flag to convert normal color palette to pastel.
- pastel_dark

  Flag to convert dark color palette to pastel.
- sat_pref_list

  List of saturation preference flags for light, normal, dark color palettes.
- ratio_dict

  A dictionary that holds the ratio of base colors in an image and is used to identify the dominant color in an image.

- base_color_dict

  A dictionary of 2D numpy arrays for each of the 6 base colors.
- extracted_colors_dict

  A Dictionary of extracted colors in [h,s,v] format.
- color_schemes_dict

  A Dictionary of dictionaries for light and dark color schemes that are in HEX string format.

6.1.1 Detailed Description

Extracts colors given a matrix of HSV values extracted from an image.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 __init__()

```
def __init__ (
        self,
        hsv_img_matrix_2d,
        output_filepath,
        pastel_light = False,
        pastel_normal = False,
        pastel_dark = False,
        sat_pref_light = False,
        sat_pref_normal = False,
        sat_pref_dark = False )
```

Extractor Constructor.

Parameters

self	The object pointer.
hsv_img_matrix_2d	A 2D numpy array of pixels from an image in [h,s,v] format.
output_filepath	Output file path with filename of where to store color palette.
pastel_light	Flag to convert light color palette to pastel.
pastel_normal	Flag to convert normal color palette to pastel.
pastel_dark	Flag to convert dark color palette to pastel.
sat_pref_light	Flag that gives preference to more saturated colors of the light color palette.
sat_pref_normal	Flag that gives preference to more saturated colors of the normal color palette.
sat_pref_dark	Flag that gives preference to more saturated colors of the dark color palette.

6.1.3 Member Function Documentation

6.1.3.1 check_pastel_conversion()

```
def check_pastel_conversion (
        self )
```

Checks to see if any of the palettes should be converted to pastel.

Parameters

self	The object pointer.

6.1.3.2 construct_scheme_dictionary()

```
def construct_scheme_dictionary (
            self )
```

Constructs a dictionary of color schemes by combining color palettes.

Light color scheme contains the normal and dark color palettes. Dark color scheme contains the normal and light color palettes.

Parameters

self	The object pointer.

6.1.3.3 convert_pastel()

```
def convert_pastel (
            self,
            hsv_color )
```

Converts/normalizes HSV color to pastel.

For values x in range [a, b], values x can be converted to the new range [y, z] with the following equation: new_x = (z-y) ∗ ((x-a) / (b-a)) + y

Parameters

self	The object pointer.
hsv_color	List HSV color to be converted to pastel.

6.1.3.4 convert_pastel_dark()

```
def convert_pastel_dark (
            self )
```

Converts dark palette to pastel.

Parameters

self	The object pointer.

6.1.3.5 convert_pastel_light()

```
def convert_pastel_light (
```

`self )`

Converts light palette to pastel.

Parameters

self	The object pointer.

6.1.3.6 convert_pastel_normal() `def convert_pastel_normal ( self )`

Converts normal palette to pastel.

Parameters

self	The object pointer.

6.1.3.7 run() `def run ( self )`

Main method for Extractor class.

Performs extraction of colors.

Parameters

self	The object pointer.

6.1.4 Member Data Documentation

6.1.4.1 base_color_dict `base_color_dict`

A dictionary of 2D numpy arrays for each of the 6 base colors.

6.1.4.2 color_schemes_dict `color_schemes_dict`

A Dictionary of dictionaries for light and dark color schemes that are in HEX string format.

6.1.4.3 extracted_colors_dict `extracted_colors_dict`

A Dictionary of extracted colors in [h,s,v] format.

6.1.4.4 hsv_img_matrix_2d `hsv_img_matrix_2d`

A 2D numpy array of pixels from an image in [h,s,v] format.

6.1.4.5 output_filepath `output_filepath`

Output file path with filename of where to store color palette.

6.1.4.6 pastel_dark `pastel_dark`

Flag to convert dark color palette to pastel.

6.1.4.7 pastel_light `pastel_light`

Flag to convert light color palette to pastel.

6.1.4.8 pastel_normal `pastel_normal`

Flag to convert normal color palette to pastel.

6.1.4.9 ratio_dict `ratio_dict`

A dictionary that holds the ratio of base colors in an image and is used to identify the dominant color in an image.

6.1.4.10 sat_pref_list `sat_pref_list`

List of saturation preference flags for light, normal, dark color palettes.

The documentation for this class was generated from the following file:

- Extractor.py

7 File Documentation

7.1 __main__.py File Reference

Main script for PyPalEx.

Namespaces

- namespace pypalex

  Python Palette Extractor: extracts color palettes from images.
- namespace pypalex.__main__

Functions

- def main ()

  Main script function.
- def handle_args ()

  Handles the arguments passed to PyPalEx.
- def extract_color_palettes ()

  Handles color extraction from image(s).
- def setup_argument_parser ()

  Sets up the argument parser for command line arguments.
- def check_sources (filepaths, path=None)

  Checks each of the sources provided and removes any bad sources.
- def check_path (path)

  Check the path to make sure it exists.
- def set_global_args (args)

  Sets the global variables using the arguments.
- def check_source (filepath)

  Checks to make sure the path leads to a file.

Variables

- list EXTRACTORS = [ ]

  List of Extractor class objects for each individual image.
- list PROPER_IMAGES = [ ]

  List of real/existing image file path(s).
- list FILENAMES = [ ]

  List of image filenames.
- list OUTPUT_FILEPATHS = [ ]

  List of output file path(s) for each image.
- string OUTPUT_PATH = ''

  The path to the output directory where all JSON files will be saved.
- string OUTPUT_TAIL = "-color_palette.json"

  The tail to append to each output filepath.
- bool PASTEL_L = False

  Flag to convert light color palette to pastel.
- bool PASTEL_N = False

Flag to convert normal color palette to pastel.
- bool PASTEL_D = False

  Flag to convert dark color palette to pastel.
- bool SAT_PREF_L = False

  Flag that gives preference to more saturated colors of the light color palette.
- bool SAT_PREF_N = False

  Flag that gives preference to more saturated colors of the normal color palette.
- bool SAT_PREF_D = False

  Flag that gives preference to more saturated colors of the dark color palette.

7.1.1 Detailed Description

Main script for PyPalEx.

Used to run from the Command Line.

7.1.2 Author(s)

- Created by Al Timofeyev on February 2, 2022.
- Modified by Al Timofeyev on April 21, 2022.
- Modified by Al Timofeyev on March 6, 2023.
- Modified by Al Timofeyev on March 22, 2023.

7.2 arg_messages.py File Reference

Archive of messages to display for arguments supplied by user.

Namespaces

- namespace pypalex

  Python Palette Extractor: extracts color palettes from images.
- namespace pypalex.arg_messages

Functions

- def bad_source_message ()

  Generates an error message if the sources provided were not images.
- def bad_path_message ()

  Generates an error message if the directory provided is not a valid directory.
- def no_args_help_message ()

  Generates a help message if no arguments were presented.

7.2.1 Detailed Description

Archive of messages to display for arguments supplied by user.

7.2.2 Author(s)

- Created by Al Timofeyev on March 3, 2022.
- Modified by Al Timofeyev on April 21, 2022.
- Modified by Al Timofeyev on March 6, 2023.

7.3 constants.py File Reference

A collection of constants for PyPalEx.

Namespaces

- namespace pypalex

  Python Palette Extractor: extracts color palettes from images.
- namespace pypalex.constants

Variables

- list BLACK_RGB = [0, 0, 0]
- list WHITE_RGB = [255, 255, 255]
- list RED_RGB = [255, 0, 0]
- list YELLOW_RGB = [255, 234, 0]
- list GREEN_RGB = [0, 255, 0]
- list CYAN_RGB = [0, 255, 255]
- list BLUE_RGB = [0, 0, 255]
- list MAGENTA_RGB = [255, 0, 255]
- int BLACK_HEX = 0x000000
- int WHITE_HEX = 0xFFFFFF
- int RED_HEX = 0xFF0000
- int YELLOW_HEX = 0xFFEA00
- int GREEN_HEX = 0x00FF00
- int CYAN_HEX = 0x00FFFF
- int BLUE_HEX = 0x0000FF
- int MAGENTA_HEX = 0xFF00FF
- int RED_HUE = 0
- int YELLOW_HUE = 55
- int GREEN_HUE = 120
- int CYAN_HUE = 180
- int BLUE_HUE = 240
- int MAGENTA_HUE = 300
- list RED_HUE_RANGE_MAX = [330, 360]
- list RED_HUE_RANGE_MIN = [0, 25]
- list YELLOW_HUE_RANGE = [25, 64]
- list GREEN_HUE_RANGE = [64, 170]
- list CYAN_HUE_RANGE = [170, 210]
- list BLUE_HUE_RANGE = [210, 260]
- list MAGENTA_HUE_RANGE = [260, 330]
- list BLACK_BRIGHTNESS_RANGE = [0.0, 50.0]
- list GRAY_BRIGHTNESS_RANGE = [50.0, 75.0]
- list WHITE_BRIGHTNESS_RANGE = [75.0, 100.0]
- list SATURATION_RANGE = [5.0, 100.0]
- list BRIGHTNESS_RANGE = [25.0, 100.0]
- list PASTEL_SATURATION_RANGE = [15.0, 75.0]
- list PASTEL_BRIGHTNESS_RANGE = [50.0, 100.0]

7.3.1 Detailed Description

A collection of constants for PyPalEx.

7.3.2 Author(s)

- Created by Al Timofeyev on February 2, 2022.
- Modified by Al Timofeyev on April 21, 2022.
- Modified by Al Timofeyev on March 6, 2023.

7.4 conversion_utils.py File Reference

Utilities for converting between RGB, HSV, HEX.

Namespaces

- namespace pypalex

  Python Palette Extractor: extracts color palettes from images.
- namespace pypalex.conversion_utils

Functions

- def rgb_to_hsv (rgb_array)

  Converts RGB array [r,g,b] to HSV array [h,s,v].
- def hsv_to_hex (hsv_array)

  Convert HSV array [h,s,v] to HEX string 'ffffff'.
- def hsv_to_rgb (hsv_array)

  Convert HSV array [h,s,v] to RGB array [r,g,b].
- def rgb_to_hex (rgb_array)

  Convert RGB array [r,g,b] to HEX string 'ffffff'.

7.4.1 Detailed Description

Utilities for converting between RGB, HSV, HEX.

7.4.2 Author(s)

- Created by Al Timofeyev on February 2, 2022.
- Modified by Al Timofeyev on April 21, 2022.
- Modified by Al Timofeyev on March 6, 2023.

7.5 extraction_utils.py File Reference

Utilities for extracting colors from the image.

Namespaces

- namespace pypalex

  Python Palette Extractor: extracts color palettes from images.
- namespace pypalex.extraction_utils

Functions

- def extract_ratios (hsv_img_matrix_2d)

  Extracts the ratios of hues per pixel.
- def construct_base_color_dictionary (hsv_img_matrix_2d)

  Constructs dictionary of base colors from an array of HSV pixel values.
- def extract_color_palettes (base_color_dict, sat_pref_list)

  Extracts dominant light, normal, dark color palettes from each of the base colors.
- def check_missing_colors (base_color_dict, extracted_colors_dict)

  Checks for any missing colors in the base color dictionary and borrows them from the surrounding colors.
- def generate_remaining_colors (extracted_colors_dict, ratios)

  Generate the remaining black and white, and background and foreground colors.
- def extract_color_types (hsv_base_color_matrix_and_sat_prefs)

  Extracts the dominant color types from a base color.
- def get_left_and_right_colors (origin_color_name)

  Gets the color names of the colors that are to the left and right of the originating color.
- def borrow_color (extracted_colors_dict, origin, borrow_left, borrow_right)

  Borrows a color from one of the extracted color types of the base colors.
- def get_dominant_hue (extracted_colors_dict, ratios)

  Calculates the dominant hue.
- def generate_black_and_white (dominant_hue)

  Generates black and white color types using the dominant hue.
- def generate_background_and_foreground (dominant_hue, complementary_hue)

  Generates the background and foreground colors.
- def sort_by_bright_value (hsv_base_color_matrix)

  Sorts the colors by the brightness value.
- def extract_dominant_color (hsv_color_type_matrix, sat_pref)

  Extracts the dominant color from a color type.
- def check_missing_color_types (light_color, norm_color, dark_color)

  Checks to make sure all the color types have been properly set.
- def check_sat_and_bright (hsv_color)

  Normalize saturation and brightness value.
- def calculate_centroid (hsv_color_type_matrix)

  Calculates the centroid for a color type.
- def find_closest_to_centroid (hsv_color_type_matrix, centroid, sat_pref)

  Finds a color from a color type that is closest to the centroid.

7.5.1 Detailed Description

Utilities for extracting colors from the image.

7.5.2 Author(s)

- Created by Al Timofeyev on February 10, 2022.
- Modified by Al Timofeyev on April 21, 2022.
- Modified by Al Timofeyev on March 6, 2023.
- Modified by Al Timofeyev on March 22, 2023.

7.6 Extractor.py File Reference

Extraction utility class for extracting colors from the image.

Classes

- class Extractor

    Extracts colors given a matrix of HSV values extracted from an image.

Namespaces

- namespace pypalex

    Python Palette Extractor: extracts color palettes from images.
- namespace pypalex.Extractor

7.6.1 Detailed Description

Extraction utility class for extracting colors from the image.

7.6.2 Author(s)

- Created by Al Timofeyev on February 10, 2022.
- Modified by Al Timofeyev on April 21, 2022.
- Modified by Al Timofeyev on March 6, 2023.
- Modified by Al Timofeyev on March 22, 2023.

7.7 image_utils.py File Reference

Utilities for processing image and file handling.

Namespaces

- namespace pypalex

  Python Palette Extractor: extracts color palettes from images.
- namespace pypalex.image_utils

Functions

- def process_image (image)

  Processes PIL Image object.
- def save_palette_to_file (color_palette, output_filepath)

  Saves color palette to json file.
- def rescale_image (image)

  Rescales image to a smaller sampling size.
- def process_helper (rgb_matrix_2d)

  Helper function for multiprocessing conversion operations.

7.7.1 Detailed Description

Utilities for processing image and file handling.

7.7.2 Author(s)

- Created by Al Timofeyev on February 27, 2022.
- Modified by Al Timofeyev on April 21, 2022.
- Modified by Al Timofeyev on March 6, 2023.

Index